How to speed up the quantization tree algorithm with an application to swing options

Anne Laure Bronstein, Gilles Pagès, Benedikt Wilbertz

How to speed up the quantization tree algorithm with an application to swing options

Anne Laure Bronstein *, Gilles Pagès † and Benedikt Wilbertz‡

November 23, 2009

Abstract

In this paper, we suggest several improvements to the numerical implementation of the quantization method for stochastic control problems in order to get fast and accurate premium estimations. This technique is applied to derivative pricing in energy markets. Several ways of modeling energy derivatives are described and finally numerical examples including parallel execution on multi-processor devices are presented to illustrate the accuracy of these methods and their execution times.

Keywords: Quantization, Gaussian process, Lévy process, Stochastic control, Swing options, Backward Dynamic programming.

MSC: 60G15, 60E99.

1 Introduction

In the last decades, optimal quantization arose as an efficient method for pricing complex financial contracts. Based on these quantizations, it is possible to construct a tree method, the quantization tree algorithm, which has successfully been applied to the pricing of derivatives like American multi-asset options (see [1]) or energy contracts like swing options (see [3] and [2]). In financial institutions, quickness of execution as well as high accuracy are important criteria in the choice of a pricing method. With this observation in mind, we suggest some improvements to the original quantization tree method. We mainly focus on the fast computations of the transition probabilities for the quantization method. In higher dimension, all procedures devoted to the computation of transition probabilities are based on Monte Carlo simulations. They all share the same core, namely repeated nearest neighbor searches. So we implemented a classical fast nearest neighbor procedure: the *kd*-tree algorithm. However, the first aim of this paper is to present and test several alternative approaches to transition probability estimation by simulation that can be massively parallelized. By "massively" we mean that the procedures can be parallelized beyond the "standard" parallelization of the plain-vanilla Monte Carlo method (consisting in generating scenarii of the underlying structure process "through" the quantization tree). These new approaches allow a bigger number of threads to run completely independent, which helps to avoid time consuming tasks like thread

*Laboratoire de Probabilités et Modèles aléatoires, UMR 7599, Université Paris 6, case 188, 4, pl. Jussieu, F-75252 Paris Cedex 5. E-mail: albronstein@gmail.com

†Laboratoire de Probabilités et Modèles aléatoires, UMR 7599, Université Paris 6, case 188, 4, pl. Jussieu, F-75252 Paris Cedex 5. E-mail: gilles.pages@upmc.fr

‡Universität Trier, FB IV-Mathematik, D-54286 Trier, Germany. E-mail: wilbertz@uni-trier.de

synchronization and communication. Some extensive numerical experiments have been carried out to compare the performances of these different approaches in term of accuracy and execution time (on both CPU and GPU).

As concerns the numerical aspects, we focused on swing options pricing, which are commonly dealt on energy markets. The underlying assets for this family of derivative products usually are forward contracts on index on gas, oil or electricity. We considered several more or less classical dynamics for the underlying asset: First some Gaussian 1-factor model, where the trinomial tree method from [14] served as reference procedure, and later on a Gaussian 2-factor model with up to 365 exercise dates. But we also considered time discretized jump dynamics, which are often used to model the spot electricity price, namely the exponential of a Normal Inverse Gaussian (NIG) Lévy process.

The NIG distribution has been introduced in finance by Barndorff-Nielsen in the 90s and has recently been applied to energy markets by several authors, e.g. Benth, Frestad and Koekebakker and Benth and Saltyte-Benth (see [5] and [6]). They suggest that the NIG family fits empirical electricity return distribution very well and represents an attractive alternative to the family of normal distributions. It is the first time to our knowledge that such a jump model is implemented in a quantization framework. Doing so, we wish to strongly emphasize the fact, that quantization tree methods are efficient to produce spatial discretization of any (simulatable) Feller Markov chain. Indeed, any (usual) time discretization scheme of a diffusion driven by a Lévy process (or any Lévy process observed at some discrete times) is a Feller Markov chain.

The third contribution of this paper is to introduce a Richardson-Romberg extrapolation based on two quantization trees of different sizes. This principle has already been successfully tested for "linear" European option pricing (see [17] or [18]), but it is the first time it is introduced in a "non linear" setting as swing option pricing and stochastic control. This idea, which is based on several heuristic considerations, seems quite promising from a numerical point of view (see section 6) and therefore strongly suggests that the theoretical expansion of the error as a function of the quantization tree size holds true.

The paper is organized as follows. In Section 2, some short background on optimal quantization is provided. In Section 3, we recall the quantization tree algorithm associated to the swing option stochastic control problem (see [3] and [2]). Moreover, a special case of swing option, the Call Strip, is developed. In Section 4, some suggestions are addressed to improve the execution time as well as the accuracy of the algorithm. Section 5 is devoted to the description of the treated dynamics and in Section 6 we finally present numerical results on both a single core and multi-core CPU resp. GPU setting.

2 Optimal quantization

Optimal quantization has been developed in the 1950s in the field of Signal Processing. Its main purpose consists in approximating a continuous signal by a discrete one in an optimal way. In the 1990s, its application has been used in the field of Numerical Integration to derive some cubature formulae (see [16]). Later in the early 2000s, this method has been extended to the computation of conditional expectations and applied to the field of Numerical Probability and Financial Mathematics. This extension has been motivated by the necessity of designing efficient methodologies for pricing and hedging more and more sophisticated financial products, especially multi-asset American options (see [1]).

Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a given probability space and let X be a random vector defined on this probability space and taking valued in $\mathbb{R}^d$. Let N be a positive integer, the first step of quantization consists in discretizing X by a $\sigma(X)$-measurable random vector $\hat{X}$ taking at most N values in a grid $\Gamma = \{x_1, \ldots, x_N\} \subset \mathbb{R}^d$. This grid or codebook is called an N-quantizer of X. Let $x = (x_1, \ldots, x_N)$

denotes the N-tuple induced by Γ. One can associate with $\hat{X}$ a Borel function $q_x : \mathbb{R}^d \to \Gamma$ called quantizer such that $\hat{X} = q_x(X)$.

Let $p \in [1, \infty[$ and let $X \in L^p(\Omega, \mathcal{A}, \mathbb{P})$. Optimal quantization consists in studying the best L^p-approximation of X by $q_x(X)$ when x runs over $(\mathbb{R}^d)^N$: it amounts to minimize in q the L^p-mean quantization error, i.e. solving the minimization problem

$$\inf\{|| X - q(X)||_p \mid q : \mathbb{R}^d \to \mathbb{R}^d \text{ Borel and } |q(X(\Omega))| \leq N\}. \tag{1}$$

Noting that

$$|X - q_x(X)| \geq \text{dist}(X, \Gamma) = \min_{1 \leq i \leq N} |X - x_i|,$$

one may restrict to a Voronoi quantization as defined below.

Definition 1. *Let $x = (x_1, \ldots, x_N) \in (\mathbb{R}^d)^N$. A partition $(C_i(x))_{1 \leq i \leq N}$ of $\mathbb{R}^d$ is a Voronoi tessellation of the N-quantizer x, if for every $i \in \{1, \ldots, N\}$, $C_i(x)$ is a Borel set satisfying*

$$C_i(x) \subset \{u \in \mathbb{R}^d \mid |u - x_i| = \min_{1 \leq j \leq N} |u - x_j|\}, \tag{2}$$

where $|.|$ is the Euclidean norm on $\mathbb{R}^d$. The nearest neighbor projection π_x on x induced by a Voronoi partition $(C_i(x))_{1 \leq i \leq N}$ is defined for every $u \in \mathbb{R}^d$ by

$$\pi_x(u) := \sum_{i=1}^{N} x_i 1_{C_i(x)}(u). \tag{3}$$

It maps the random vector X into

$$\hat{X}^x = \sum_{i=1}^{N} x_i 1_{C_i(x)}(X) = \pi_x(X), \tag{4}$$

which is called a Voronoi quantization of X.

Now, the minimization problem (1) turns into an optimization problem on $x \in (\mathbb{R}^d)^N$. This second step depends on p and the probability distribution $\mathbb{P}_X$ of X. It always has at least one solution x^* (see e.g. [12] or [16]). If moreover $\text{card}(\text{supp}\, \mathbb{P}_X) = \infty$, then x^* has pairwise distinct components and $\min_{x \in (\mathbb{R}^d)^N} || X - \hat{X}^x||_p$ is decreasing to 0 as N goes to ∞.

To be more precise, the Zador Theorem (see [12]) provides a rate for the optimal L^p-mean quantization error, namely, if $X \in L^{p'}$ for some $p' > p$,

$$\min_{x \in (\mathbb{R}^d)^N} || X - \hat{X}^x||_p = \mathcal{O}(N^{-1/d}) \quad \text{as } N \to \infty. \tag{5}$$

We refer to the following papers for a study of several algorithms designed to find optimal quantizers: (see [15, 11, 16, 17]). The problem dimension and the properties of the law of X might help to determine an efficient algorithm. Note that some optimized quantizers of the normal distribution $\mathcal{N}(0, I_d)$ have been computed and are available at the URL

`www.quantize.math-fi.com`

for dimensions up to 10 and sizes up to several thousands.

Since an optimal quantization $\hat{X}^x$ provides the best finite approximation to the distribution of X in the least square sense, it becomes natural to use $\mathbb{E}f(\hat{X}^x)$ as an approximation for $\mathbb{E}f(X)$, where $f : \mathbb{R}^d \to \mathbb{R}$ is a Borel function.

Note further, that since $\hat{X}^x$ takes only finitely many values, we compute $\mathbb{E}f(\hat{X}^x)$ as the finite sum

$$\sum_{i=1}^{N} \mathbb{P}(X \in C_i(x)) f(x_i).$$

The weights $p_i := \mathbb{P}(X \in C_i(x))$ of this cubature formula are obtained as a by-product of optimization procedures to generate optimal quantizers like the ones used in [17] or [16].

Assume now that f exhibits some smoothness properties, i.e. f is differentiable with Lipschitz continuous differential Df. If x is a stationary quantizer, i.e.

$$\hat{X}^x = \mathbb{E}(X|\hat{X}^x),$$

(so is the case for every optimal L^2-quantizer), we can conclude from a second order Taylor expansion of f, that the approximation error of $\mathbb{E}f(\hat{X}^x)$ satisfies

$$|\mathbb{E}f(X) - \mathbb{E}f(\hat{X}^x)| \leq [Df]_{\mathrm{Lip}} \|X - \hat{X}^x\|_2^2.$$

Similarly, we can derive (see e.g. [18]) some error bounds for the approximation of $\mathbb{E}(f(X)|Y)$ by its optimal quantized counterpart $\mathbb{E}(f(\hat{X}^x)|\hat{Y}^y)$.

3 The quantization tree algorithm

3.1 Abstract problem formulation

We briefly recall the general stochastic control framework developed in [2] to model "abstract" swing options and the quantization tree algorithm devised to solve it.

Let $(t_k)_{0\leq k\leq n-1}$ be a given, strictly increasing sequence of exercise dates, such that the first exercise date is set at the origin of the derivative and the last one happens strictly before the option maturity.

Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a complete probability space and let $p \in [1, \infty[$. On this probability space, one defines an option with maturity T by a sequence of random variables $(V_k)_{0\leq k\leq n-1}$. For $k \in \{0, \ldots, n-1\}$, V_k represents the option payoff obtained by the derivative holder at time t_k. To model all the available market information, one introduces a filtration $\mathcal{F} := (\mathcal{F}_k)_{0\leq k\leq n-1}$ on $(\Omega, \mathcal{A}, \mathbb{P})$, such that, for $k \in \{0, \ldots, n-1\}$, the option payoff V_k is measurable with respect to the σ-field $\mathcal{F}_k$. One common feature characterizing most of the options is that the product is built on an underlying asset. In our setting, the option owner has the possibility to choose, for each exercise date, the underlying asset amount included in the contract. These decision variables are represented by a sequence $(q_k)_{0\leq k\leq n-1}$ of $\mathcal{F}$-adapted random variables and are subjected to constraints given by:

(i) a local constraint:
let $q_{\min}$ and $q_{\max}$ be two given positive constants. For $k = 0, \ldots, n-1$,

$$q_{\min} \leq q_k \leq q_{\max} \quad \mathbb{P}\text{-a.s..} \tag{6}$$

(ii) A global constraint:
one defines a cumulative purchased process $(\bar{q}_k)_{0\leq k\leq n}$ by

$$\bar{q}_0 := 0 \quad \text{and} \quad \bar{q}_k := \sum_{l=0}^{k-1} q_l, \quad k = 1, \ldots, n.$$

Then for any couple of $\mathcal{F}_0$-measurable, non-negative random variables $Q := (Q_{\min}, Q_{\max})$, the total cumulative volume purchased should satisfy

$$Q_{\min} \leq \bar{q}_n \leq Q_{\max} \quad \mathbb{P}\text{-a.s..} \tag{7}$$

A control process $(q_k)_{0\leq k\leq n-1}$ which satisfies these local and global conditions is called $(\mathcal{F}, Q)$-admissible.

Now, one defines the residual global constraints $Q^k = (Q^k_{\min}, Q^k_{\max})$ at time t_k by

$$Q^k_{\min} = Q_{\min} - \bar{q}_k \quad \text{and} \quad Q^k_{\max} = Q_{\max} - \bar{q}_k.$$

Then, the associated option value at time t_k is given by

$$P^n_k((Q^k_{\min}, Q^k_{\max})) := \operatorname{esssup}\left\{\mathbb{E}\left(\sum_{l=k}^{n-1} q_l V_l | \mathcal{F}_k\right), q_l : (\Omega, \mathcal{F}_l) \to [q_{\min}, q_{\max}], k \leq l \leq n-1, \bar{q}_n - \bar{q}_k \in [Q^k_{\min}, Q^k_{\max}]\right\}. \tag{8}$$

Now, to simplify notations, we perform a decomposition of the problem into a simple "swap" part and a "normalized" one, where the control variables are $[0,1]$ valued, i.e. $q = q_{\min} + (q_{\max} - q_{\min})q'$ for a $[0,1]$ valued r.v. q', which leads to

$$P^n_k((Q^k_{\min}, Q^k_{\max})) = q_{\min} \sum_{l=k}^{n-1} \mathbb{E}\left(V_l | \mathcal{F}_k\right) + (q_{\max} - q_{\min}) P^{[0,1],n}_k((\tilde{Q}^k_{\min}, \tilde{Q}^k_{\max})) \tag{9}$$

with

$$\tilde{Q}^k_{\min} = \frac{Q^k_{\min} - (n-k)q_{\min}}{q_{\max} - q_{\min}} \quad \text{and} \quad \tilde{Q}^k_{\max} = \frac{Q^k_{\max} - (n-k)q_{\min}}{q_{\max} - q_{\min}}.$$

One considers from now on a normalized framework, i.e. $q_{\min} = 0$ and $q_{\max} = 1$. In this setting, a couple of global constraints Q^k at time t_k is admissible if Q^k is $\mathcal{F}_k$-measurable and if

$$0 \leq Q^k_{\min} \leq Q^k_{\max} \leq n-k \quad \mathbb{P}\text{-a.s..} \tag{10}$$

For every couple of admissible global constraints Q^k, the derivative price at time t_k satisfies

$$P^n_k((Q^k_{\min}, Q^k_{\max})) := \operatorname{esssup}\left\{\mathbb{E}\left(\sum_{l=k}^{n-1} q_l V_l | \mathcal{F}_k\right), q_l : (\Omega, \mathcal{F}_l) \to [0,1], k \leq l \leq n-1, \bar{q}_n - \bar{q}_k \in [Q^k_{\min}, Q^k_{\max}]\right\}. \tag{11}$$

To design the quantization tree algorithm, we appeal to several results summarized below (see [2] for further details):

(i) Without loss of generality, one may assume that at time t_k the couple of admissible global constraints Q^k is deterministic.

(ii) The Backward Dynamic Programming Formula (BDP):

Set $P^n_n \equiv 0$.

For every $k \in \{0, \ldots, n-1\}$ and every couple of admissible global constraints $Q^k = (Q^k_{\min}, Q^k_{\max})$ at time t_k, we have

$$P^n_k(Q^k) = \sup\left\{xV_k + \mathbb{E}(P^n_{k+1}(\chi^{n-k-1}(Q^k, x))|\mathcal{F}_k), x \in I^{n-k-1}_{Q^k}\right\}$$

with

$$\chi^M(Q^k, x) := \left((Q^k_{\min} - x)^+, (Q^k_{\max} - x) \wedge M\right)$$

and

$$I^M_{Q^k} := [(Q^k_{\min} - M)^+ \wedge 1, Q^k_{\max} \wedge 1].$$

(iii) *Bang-bang control: One defines the set of admissible deterministic global constraints at time 0 by*

$$T^+(n) = \{Q^0 \in \mathbb{R}^2 \mid 0 \leq Q^0_{min} \leq Q^0_{max} \leq n\}. \tag{12}$$

Then, given any admissible integral global constraints at time 0 *i.e.* $Q^0 \in \mathbb{N}^2 \cap T^+(n)$*, there exists an optimal bang-bang control process* $q^* = (q^*_k)_{0\leq k\leq n-1}$ *with* $q^*_k \in \{0,1\}$ $\mathbb{P}$*-a.s. for every* $k = 0, \ldots, n-1$*.*

(iv) The value function $Q^0 \mapsto P^n_0(Q^0)$ *is concave, continuous and piecewise affine on the tiling of* $T^+(n)$*.*

So in view of these results, it is sufficient to evaluate the option premium at integral values of $T^+(n)$. Then, it is possible to use a linear interpolation inside every tile to obtain the derivative price for all admissible constraints, i.e. for all $Q^0 \in T^+(n)$. Hence we may reformulate the BDP Formula for an initial $Q^0 \in \mathbb{N}^2 \cap T^+(n)$ as

$$\begin{aligned} P^n_n &\equiv 0 \\ P^n_k(Q^k) &= \max\left\{xV_k + \mathbb{E}(P^n_{k+1}(\chi^{n-k-1}(Q^k,x))|\mathcal{F}_k), x \in \{0,1\} \cap I^{n-k-1}_{Q^k}\right\}, \qquad k = 0,\ldots,n-1. \end{aligned} \tag{13}$$

In order to simulate the derivative prices obtained in (13), one appeals to the quantization method described in Section 2. At each exercise date t_k, $k = 0, \ldots, n-1$, we perform a spatial discretization of the d-dimensional underlying asset and include the resulting discretized process in the BDP Formula. Finally, some convergence theorems are available to obtain some error bounds.

However, these theorems require the following assumption:
Assume that there exists a d-dimensional Markov structure process, say $(X_k)_{0\leq k\leq n-1}$, such that the payoffs V_k are function of X_k, i.e.

$$V_k = v_k(X_k), \quad k = 0, \ldots, n-1. \tag{14}$$

One obtains

$$\mathbb{E}(v_{k+1}(X_{k+1})|\mathcal{F}_k) = \mathbb{E}(v_{k+1}(X_{k+1})|X_k) = \Theta_k(v_{k+1})(X_k), \quad k = 0, \ldots, n-1, \tag{15}$$

where $(\Theta_k)_{0\leq k\leq n-1}$ is a sequence of Borel transition probabilities on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$. Therefore (13) becomes

$$\begin{aligned} P^n_n &\equiv 0 \\ P^n_k(Q^k) &= \max\left\{xv_k(X_k) + \mathbb{E}(P^n_{k+1}(\chi^{n-k-1}(Q^k,x))|X_k), x \in \{0,1\} \cap I^{n-k-1}_{Q^k}\right\}, \qquad k = 0,\ldots,n-1. \end{aligned} \tag{16}$$

Now, let $\hat{X}_k$ be a quantized version of X_k of size say N_k. An approximation of the price process is designed by plugging $\hat{X}_k$ in (16) and by *forcing the Markov property* on $\hat{X}_k$.

$$\begin{aligned} \hat{P}^n_n &\equiv 0 \\ \hat{P}^n_k(Q^k) &= \max\left\{xv_k(\hat{X}_k) + \mathbb{E}(\hat{P}^n_{k+1}(\chi^{n-k-1}(Q^k,x))|\hat{X}_k), x \in \{0,1\} \cap I^{n-k-1}_{Q^k}\right\}, \qquad k = 0,\ldots,n-1. \end{aligned} \tag{17}$$

The resulting algorithm, called quantization tree, is then tractable on a computer. Indeed, it is possible to evaluate the above formula in a recursive backward way down to $k = 0$ where we arrive at a $\sigma(X_0)$-measurable solution. Note that if $\mathcal{F}_0 = \{\emptyset, \Omega\}$, the option premium is deterministic. From now on, we assume that $\mathcal{F}_0$ is trivial.

The following results are concerned with the error resulting from the discretization of the underlying price process $(X_k)_{0\leq k\leq n-1}$ (see [2]).

Theorem 3.1. *Assume that the Markov process $(X_k)_{0\leq k\leq n-1}$ is Lipschitz Feller in the following sense: for every bounded Lipschitz continuous function $g : \mathbb{R}^d \to \mathbb{R}$ and every $k \in \{0, \dots, n-1\}$, $\Theta_k(g)$ is a Lipschitz continuous function satisfying $[\Theta_k(g)]_{Lip} \leq [\Theta_k]_{Lip}[g]_{Lip}$. Assume that every function $v_k : \mathbb{R}^d \to \mathbb{R}$ is Lipschitz continuous with Lipschitz coefficient $[v_k]_{Lip}$. Let $p \in [1, \infty[$ such that $\max_{0\leq k\leq n-1} |X_k| \in L^p(\mathbb{P})$. Then, there exists a real constant $C_p > 0$ such that*

$$\| \sup_{Q\in T^+(n)\cap\mathbb{N}^2} |P_0^n(Q) - \hat{P}_0^n(Q)|\|_p \leq C_p \sum_{k=0}^{n-1} \|X_k - \hat{X}_k\|_p.$$

In view of Zador's Theorem (5) and the $\mathcal{F}_0$-measurability of P_0^n and $\hat{P}_0^n$, this also reads

$$\sup_{Q\in T^+(n)\cap\mathbb{N}^2} |P_0^n(Q) - \hat{P}_0^n(Q)| \leq C \sum_{k=0}^{n-1} N_k^{-1/d}.$$

3.2 Complexity and implementationary notes

For an analysis of the complexity of the quantization tree algorithm in its original form (17), we count the number of multiplications, which occur during the evaluation of $P_0^n(Q^0)$ for a given Q^0. We implement the quantization tree algorithm in a backward iterative manner, starting at layer k with the computation of $P_k^n(Q^k)$ for every possible residual global constraint $Q^k \in \mathcal{Q}_k^n(Q^0)$ with

$$\mathcal{Q}_k^n(Q^0) = \left\{ \left((Q_{\min}^0 - l)^+, (Q_{\max}^0 - l)^+ \wedge (n-k) \right), l = 0, \dots, k \right\}$$

for given initial global constraints $Q = (Q_{\min}, Q_{\max}) \in T^+(n) \cap \mathbb{N}^2$. This approach yields a complexity proportional to

$$\sum_{k=0}^{n-2} \#\mathcal{Q}_k^n(Q^0) N_k N_{k+1} + N_{n-1}$$

multiplications, which can be estimated from above by

$$\sum_{k=0}^{n-2} (k+1) N_k N_{k+1} + N_{n-1}.$$

If we employ an equal grid size N in each layer for the quantization of the conditional expectations, we finally arrive at an upper bound proportional to $\frac{n^2N^2}{2}$ multiplications.

3.3 The call strip: case $Q = (0, n)$

We will study a special case of swing options. This class is characterized by particular values of global constraints.

In this case the global constraint is always satisfied by the cumulative purchase process, i.e. $\sum_{l=0}^{n-1} q_l \in [Q_{\min}, Q_{\max}]$, as the control process $(q_l)_{0\leq l\leq n-1}$ is $[0, 1]$-valued. Hence, the optimal

control problem is reduced to a maximization problem without constraints:

$$P_k^n(0,n) = \operatorname{esssup}\left\{ \mathbb{E}\left(\sum_{l=k}^{n-1} q_l V_l | \mathcal{F}_k \right), q_l : (\Omega, \mathcal{F}_l) \to [0,1], k \le l \le n-1 \right\}, \qquad k = 0, \dots, n-1. \tag{18}$$

An optimal control for (18) is clearly given by

$$q_l^* := \begin{cases} 1 & \text{on } \{V_l \ge 0\}, \\ 0 & \text{elsewhere,} \end{cases}$$

and the associated price process by

$$P_k^n(0,n) = \sum_{l=k}^{n-1} \mathbb{E}\left({V_l}^+ | \mathcal{F}_k \right).$$

Calling upon the results of Section (3.1), one obtains the following quantized tree algorithm in a Markovian structure

$$\hat{P}_k^n(0,n) = \sum_{l=k}^{n-1} \mathbb{E}\left((v_l(\hat{X}_l))^+ | \hat{X}_k \right).$$

Note that this quantity appears as a sum of European options. Since closed forms are available to evaluate European options, the call strip will be used as benchmark for numerical implementations.

Remark 1. *The case $Q = (0,1)$ leads in the same way to the payoff of a Bermudan option.*

4 Two main improvements for the quantization tree algorithm

4.1 Computations of the transition probabilities: fast nearest neighbor search

Solving the quantized BDP principle (17) amounts from a numerical point of view to compute for some Borel function $f : \mathbb{R}^d \to \mathbb{R}$ all the conditional expectations

$$\mathbb{E}(f(\hat{X}_{k+1}) | \hat{X}_k),$$

which take values in the finite grid $x^k = (x_1^k, \dots, x_{N_k}^k)$ of $\mathbb{R}^d$. Moreover, we derive that

$$\mathbb{E}(f(\hat{X}_{k+1}) | \hat{X}_k = x_i^k) = \sum_{j=1}^{N_{k+1}} f(x_j^{k+1}) \pi_k^{ij}$$

where

$$\pi_k^{ij} = \mathbb{P}(\hat{X}_{k+1} = x_j^{k+1} | \hat{X}_k = x_i^k), \quad k = 0, \dots, n-2,$$

denote the transition probabilities of the quantized process. By definition of the Voronoi quantization $\hat{X}$ in (4), we arrive at

$$\pi_k^{ij} = \mathbb{P}(X_{k+1} \in C_j(x^{k+1}) | X_k \in C_i(x^k)) = \frac{\mathbb{P}(X_{k+1} \in C_j(x^{k+1}), X_k \in C_i(x^k))}{\mathbb{P}(X_k \in C_i(x^k))}. \tag{19}$$

To estimate the transition probabilities by means of a naive Monte Carlo simulation, one generates samples of the Markov chain $(X_k)_{0 \le k \le n-1}$ with size say M, which we denote by $(\tilde{X}_k^m)_{1 \le m \le M}$, $k = 0, \dots, n-1$. So π_k^{ij} can be estimated by

$$\tilde{\pi}_k^{ij} := \frac{\sum_{m=1}^M 1_{C_j(x^{k+1})}(\tilde{X}_{k+1}^m) 1_{C_i(x^k)}(\tilde{X}_k^m)}{\sum_{m=1}^M 1_{C_i(x^k)}(\tilde{X}_k^m)}, \quad \text{for } k = 0, \dots, n-2, \tag{20}$$
$$i = 1, \dots, N_k \text{ and } j = 1, \dots, N_{k+1}.$$

Given k, i and j, one has to check for each $m = 1, \dots, M$ if the realizations $\tilde{X}_k^m(\omega)$ and $\tilde{X}_{k+1}^m(\omega)$ belong to the cells $C_i(x^k)$ or $C_j(x^{k+1})$ of the Voronoi tessellation to determine the value of the indicator functions present in the estimations (20).

Nearest neighbor search: This problem consists in finding the nearest neighbor for a query point $q \in \mathbb{R}^d$ in the N-tuple $x = (x_1, \dots, x_N)$ with regard to the Euclidean distance. A commonly known algorithm to optimize this nearest neighbor search, which in high dimensions can become a time consuming procedure, is the *kd*-tree algorithm (see e.g. [8]).

This procedure consists of two steps: The partitioning of the space, which has an average complexity cost of $\mathcal{O}(n \log n)$ operations. This has to be done only once and is therefore independent of the size M of the Monte Carlo sample. The second part is the search procedure, which has an average complexity cost of $\mathcal{O}(\log n)$ distance computations.

The *kd*-tree data structure is based on a recursive subdivision of space $\mathbb{R}^d$ into disjoint hyper-rectangular regions called boxes. Each node of the resulting tree is associated with such a region. To ensure a well-defined splitting rule on each node, we equip the root node with a bounding box, which is chosen to contain all data points.

As long as the number of data points within such box is greater than a given threshold, the box is split-up into two boxes by an axis-orthogonal hyperplane that intersects the original box. These two boxes are then associated to two children nodes. When the number of points in a box is below the threshold, the associated node is declared a leaf node, and the points are stored within this node.

To find the nearest neighbor for a query point q, we start at the root node and decide at each node due to the cutting hyperplane, to which child node we descend.

When arriving at a leaf node, we compute the minimal distance from all data points of this node to the query point q and ascend back on the trial to the root node as long as it may be possible to find closer points than the previous found minimal distance. During this way back we obviously descend to the so far not visited children and update the minimal distance found so far.

From now on, we assume for convenience that $(X_k)_{0 \le k \le n-1}$ can be written recursively, i.e.

$$X_{k+1} = A_k X_k + T_k Z_k, \quad k = 0, \dots, n-2, \qquad X_0 = x_0, \tag{21}$$

for some positive definite Matrices A_k and T_k, and a deterministic initial value x_0. Furthermore we suppose, that the distribution of Z_k is either known through its density or can be simulated at some reasonable costs and that Z_k is independent of the distribution of X_l, $l = 0, \dots, k$.

These assumptions hold for example for the Black-Scholes model (where the driving process is a Brownian motion), the Ornstein-Uhlenbeck process, which we will later on represent as a Gaussian first order auto-regressive process or for certain Lévy processes, e.g. the NIG process.

4.2 Parallelization of the transition probability estimation

4.2.1 Diffusion method

Using the representation (21) of X as an auto-regressive process, the naive Monte Carlo approach from (20) now reads as follows. Generate a Monte Carlo sample of size M of the random variate Z_k in each layer k, i.e

$$(\tilde{Z}_0^m)_{1 \le m \le M}, \dots, (\tilde{Z}_{n-2}^m)_{1 \le m \le M},$$

and define the Monte Carlo sample for $(X_k)_{0 \le k \le n-1}$ by

$$\begin{aligned} \tilde{X}_0^m &:= x_0, \\ \tilde{X}_{k+1}^m &:= A_k \tilde{X}_k^m + T_k \tilde{Z}_k^m, \qquad k = 0, \dots, n-2 \end{aligned} \tag{22}$$

for $m = 1, \ldots, M$. Then we get an estimate for π_k^{ij}, which writes

$$\tilde{\pi}_k^{ij} := \frac{\sum_{m=1}^M 1_{C_j(x^{k+1})}(\tilde{X}_{k+1}^m) 1_{C_i(x^k)}(\tilde{X}_k^m)}{\sum_{m=1}^M 1_{C_i(x^k)}(\tilde{X}_k^m)}.$$

It is possible to implement a parallel procedure of the diffusion method like for any "linear" Monte Carlo simulation. However the computational structure of the transition probabilities is not well adapted to this parallel implementation. Indeed, in this method, we have to proceed the computations along the discrete trajectories of $\tilde{X}^m$, that is, step by step from time 0 up to the exercise date t_{n-1}. Therefore, the natural way to partition the tasks would be to divide the number of Monte Carlo simulations into several processes. Once the computations executed simultaneously by each process are completed and sent back to the main process, this process still would have to add up the partial sums for each transition in each layer k, which leads to some additional and significant work. So, a more advantageous approach from the parallelization viewpoint will be described in the section below.

4.2.2 Parallel Quantization Weight Estimation

This method was introduced in [3] and heavily relies on the assumptions on X satisfying the auto-regressive structure (21). We generate in each layer k, independently, bivariate Monte Carlo samples by simulating directly from the distributions of X_k and Z_k, i.e.

$$(\tilde{X}_0^m, \tilde{Z}_0^m)_{1 \le m \le M}, \ldots, (\tilde{X}_{n-2}^m, \tilde{Z}_{n-2}^m)_{1 \le m \le M}.$$

The estimated transition probabilities are then given by

$$\tilde{\pi}_k^{ij} := \frac{\sum_{m=1}^M 1_{C_j(x^{k+1})}(\tilde{A}_k \tilde{X}_k^m + T_k \tilde{Z}_k^m) 1_{C_i(x^k)}(\tilde{X}_k^m)}{\sum_{m=1}^M 1_{C_i(x^k)}(\tilde{X}_k^m)}.$$

In this approach, the computations of the transitions for a layer k are completely independent from all the preceding and succeeding layers. So a parallel implementation with respect to the time layers t_k becomes very straightforward. Additionally, all the ideas for a further parallel split-up with regard to the number of Monte Carlo simulations, as for the Diffusion method, remain valid.

4.2.3 Spray method

The spray method is not an exact approach, since it aims at estimating only an approximation of π_k^{ij}. This means that we not only replace the random variable X_k by its quantization $\hat{X}_k$ in the identity $X_{k+1} = A_k X_k + T_k Z_k$, but also in the conditional part $X_k \in C_i(x^k)$ of (19).

Using the equivalence of $\hat{X}_k \in C_i(x^k)$ and $\hat{X}_k = x_i^k$, this leads to the following approximation

$$\begin{aligned} \tilde{\pi}_k^{ij} &:= \mathbb{P}(A_k \hat{X}_k + T_k Z_k \in C_j(x^{k+1}) | \hat{X}_k \in C_i(x^k)) \\ &= \mathbb{P}(A_k x_i^k + T_k Z_k \in C_j(x^{k+1})) \\ &= \mathbb{P}\left(Z_k \in T_k^{-1}(C_j(x^{k+1}) - A_k x_i^k)\right). \end{aligned}$$

In low dimensions and when the density of Z_k is known, this quantity can be computed by deterministic integration methods. Otherwise a Monte Carlo estimate would be given by

$$\tilde{\pi}_k^{ij} := \frac{1}{M} \sum_{m=1}^M 1_{T_k^{-1}(C_j(x^{k+1}) - A_k x_i^k)}(\tilde{Z}_k^m),$$

where

$$(\tilde{Z}_0^m)_{1\le m\le M},\dots,(\tilde{Z}_{n-2}^m)_{1\le m\le M}$$

denote again the i.i.d. Monte Carlo samples of (Z_k).

Note that such an approach can be extended to a non linear dynamic $X_{k+1}=F(X_k,Z_k),\ Z_k$ i.i.d..

4.3 Improving the convergence rate: a Romberg extrapolation approach

Consider for instance the classical numerical integration by means of optimal quantization, i.e. $\mathbb{E}f(X)$ is approximated by $\mathbb{E}f(\hat{X}^N)$ for sequence of stationary and rate optimal quantizers $\hat{X}^N$ (see Section 2).

Assume that $f:\mathbb{R}^d\to\mathbb{R}$ is a 3-times differentiable function with bounded derivatives.

We also assume (which is still a conjecture) that

$$\mathbb{E}(D^2f(\hat{X}^N)(X-\hat{X}^N)^{\otimes 2})\sim cN^{-2/d}\qquad \text{as } N\to\infty$$

and that

$$\mathbb{E}|X-\hat{X}^N|^3=\mathcal{O}(N^{-3/d}).$$

Then we can conclude from a Taylor expansion

$$f(X)=f(\hat{X}^N)+Df(\hat{X}^N)(X-\hat{X}^N)+\frac{1}{2}D^2f(\hat{X}^N)(X-\hat{X}^N)^{\otimes 2}+\frac{1}{6}D^3f(\xi)(X-\hat{X}^N)^{\otimes 3},\qquad \xi\in(\hat{X}^N,X),$$

so that

$$\mathbb{E}f(X)=\mathbb{E}f(\hat{X}^N)+cN^{-2/d}+o(N^{-3/d+\varepsilon})\qquad \text{for every } \varepsilon>0.$$

Motivated by this identity and our numerical tests, we make the conjecture, that an analogous result holds also (at least approximately) true for the quantization tree algorithm

$$P_0^n=\hat{P}_0^n+cN^{-2/d}+o(N^{-3/d+\varepsilon})\qquad \text{for every } \varepsilon>0.$$

Hence we perform a Romberg extrapolation on the quantized swing option prices by computing $\hat{P}_0^n$ for two different values N_1,N_2 (e.g. $N_1\approx 4N_2$) and denote them by $\hat{P}_0^n(N_1)$ and $\hat{P}_0^n(N_2)$. Thus we arrive at

$$P_0^n=\hat{P}_0^n(N_1)+\frac{\hat{P}_0^n(N_1)-\hat{P}_0^n(N_2)}{N_2^{-2/d}-N_1^{-2/d}}N_1^{-2/d}+o(N^{-3/d+\varepsilon}).$$

This suggests to consider as improved approximation for P_0^n the quantity

$$P_0^n(N_1,N_2):=\hat{P}_0^n(N_1)+\frac{\hat{P}_0^n(N_1)-\hat{P}_0^n(N_2)}{N_2^{-2/d}-N_1^{-2/d}}N_1^{-2/d}.$$

5 Application: swing options

We will consider in the section three different ways of modeling for the underlying dynamics, which all fulfill the assumptions of the quantization tree algorithm. These are namely the 1- and 2-factor Gaussian model and an exponential Lévy model.

From now on, we assume an equidistant time discretization $0=t_0<t_1<\ldots<t_n=T$, i.e. $t_k=k\Delta t, k=0,\dots,n$ with $\Delta t:=T/n$.

5.1 Gaussian 1-factor model

In this case, we consider a dynamic of the underlying forward curve $(F_{t,T})_{t\in[0,T]}$ given by the SDE

$$dF_{t,T} = \sigma e^{-\alpha(T-t)} F_{t,T} dW_t,$$

which yields

$$S_t = F_{0,t} \exp(\sigma \int_0^t e^{-\alpha(t-s)} dW_s - \frac{1}{2}\Delta_t^2)$$

for

$$\Delta_t^2 = \frac{\sigma^2}{2\alpha}\left(1 - e^{-2\alpha t}\right).$$

If we set $X_k := \int_0^{k\Delta t} e^{-\alpha(k\Delta t - s)} dW_s, k = 0, \ldots, n-1$, $(X_k)_{0\le k\le n-1}$ is a discrete time Markov process and we get $S_{k\Delta t} - K = v_k(X_k)$ with Lipschitz-continuous $v_k(x) = (F_{0,k\Delta t} \exp(\sigma x - \frac{1}{2}\Delta_{k\Delta t}^2) - K)$. So the formal requirements for the application of the quantization tree algorithm are fulfilled. For the quantization and the computation of the transition probabilities of X_k we need the following results:

Proposition 5.1. *The discrete time Ornstein-Uhlenbeck process $X_k = \int_0^{k\Delta t} e^{-\alpha(k\Delta t - s)} dW_s$ can be written as first order auto-regressive process*

$$X_{k+1} = e^{-\alpha\Delta t} X_k + \sqrt{\frac{1 - e^{-2\alpha\Delta t}}{2\alpha}} \epsilon_k, \qquad k = 0, \ldots, n-2, \quad X_0 := 0,$$

for an i.i.d sequence (ϵ_k), $\epsilon_1 \sim \mathcal{N}(0,1)$.

Especially we have

$$X_k \sim \mathcal{N}\left(0, \frac{1 - e^{-2\alpha t_k}}{2\alpha}\right).$$

Since affine transformations of a one-dimensional random variate can be transformed one-to-one on its optimal quantizers, the quantizers for X_k can be constructed as a dilatation of optimal quantizers for $\mathcal{N}(0,1)$ by the factor $\sqrt{\frac{1-e^{-2\alpha t_k}}{2\alpha}}$.

5.2 Gaussian 2-factor model

Furthermore we have also considered a Gaussian 2-factor model, where the dynamic of the forward curve $(F_{t,T})_{t\in[0,T]}$ is given by the SDE

$$dF_{t,T} = F_{t,T}\left(\sigma_1 e^{-\alpha_1(T-t)} dW_t^1 + \sigma_2 e^{-\alpha_2(T-t)} dW_t^2\right)$$

for two Brownian motions W^1 and W^2 with correlation coefficient ρ.

This yields

$$S_t = F_{0,t} \exp\left(\sigma_1 \int_0^t e^{-\alpha_1(t-s)} dW_s^1 + \sigma_2 \int_0^t e^{-\alpha_2(t-s)} dW_s^2 - \frac{1}{2}\Delta_t^2\right)$$

for

$$\Delta_t^2 = \frac{\sigma_1^2}{2\alpha_1}\left(1 - e^{-2\alpha_1 t}\right) + \frac{\sigma_2^2}{2\alpha_2}\left(1 - e^{-2\alpha_2 t}\right) + 2\rho\frac{\sigma_1\sigma_2}{\alpha_1 + \alpha_2}\left(1 - e^{-(\alpha_1+\alpha_2)t}\right).$$

In this case we have to choose

$$X_k := \left(\int_0^{k\Delta t} e^{-\alpha_1(k\Delta t - s)} dW_s^1, \int_0^{k\Delta t} e^{-\alpha_2(k\Delta t - s)} dW_s^2\right), k = 0, \ldots, n-1,$$

as underlying Markov structure process with

$$v_k(x_1, x_2) = (F_{0,k\Delta t} \exp(\sigma_1 x_1 + \sigma_2 x_2 - \frac{1}{2}\Delta_{k\Delta t}^2) - K).$$

Applying Proposition 5.1 on the two components of $(X_k)_{0\leq k\leq n-1}$ allows us to write it as a first order auto-regressive process

$$X_{k+1} = A_k X_k + T_k \epsilon_k$$

with

$$A = \begin{pmatrix} e^{-\alpha_1 \Delta t} & 0 \\ 0 & e^{-\alpha_2 \Delta t} \end{pmatrix}$$

and

$$T = \begin{pmatrix} \sqrt{\frac{1}{2\alpha_1}(1 - e^{-2\alpha_1 \Delta t})} & 0 \\ \sqrt{\frac{1}{2\alpha_2}(1 - e^{-2\alpha_2 \Delta t})} r & \sqrt{\frac{1}{2\alpha_2}(1 - e^{-2\alpha_2 \Delta t})}\sqrt{1 - r^2} \end{pmatrix}$$

where

$$r = \rho \frac{\frac{1}{\alpha_1 + \alpha_2}\left(1 - e^{-(\alpha_1 + \alpha_2)\Delta t}\right)}{\sqrt{\frac{1}{4\alpha_1 \alpha_2}(1 - e^{-2\alpha_1 \Delta t})(1 - e^{-2\alpha_2 \Delta t})}}.$$

5.3 Normal Inverse Gaussian spot return

In this case we assume, that the dynamics of the underlying is driven by the exponential of a special Lévy process, the so-called *Normal Inverse Gaussian* process (NIG). This means, that we assume, that the dynamic of the underlying is given by

$$S_t = S_0 \exp(L_t)$$

for a NIG process $(L_t)_{t\in[0,T]}$. This special model for financial data has been first proposed by Barndorff-Nielsen in [4] and has been later applied to financial contracts on energy markets (see e.g. [5] and [6]) as well.

The NIG process has beneath its Lévy property, which makes it a Markov process, the convenient property, that it is completely determined by its distribution at time t =1, the so-called NIG distribution, NIG$(\alpha, \beta, \delta, \mu)$, with parameters $\alpha > 0, |\beta| < \alpha, \delta > 0$ and $\mu \in \mathbb{R}$, i.e.

$$L_1 \sim \text{NIG}(\alpha, \beta, \delta, \mu) \qquad \text{and} \qquad L_t \sim \text{NIG}(\alpha, \beta, t\delta, t\mu)$$

Thus, as soon as we have knowledge on some properties of the NIG distribution like the density or the characteristic function, we know it already for $(L_t)_{t\in[0,T]}$ at any timescale.

So, the density of $(L_t)_{t\in[0,T]}$ is for example given by

$$f_t^{\text{NIG}(\alpha,\beta,\delta,\mu)}(x) = \alpha\delta t e^{t\delta\sqrt{\alpha^2-\beta^2}+\beta(x-t\mu)} \frac{K_1(\alpha\sqrt{t^2\delta^2 + (x-t\mu)^2})}{\pi\sqrt{t^2\delta^2 + (x-t\mu)^2}},$$

where K_1 denotes the Bessel function of third kind with index 1.

These facts will help to the computation of optimal quantizers for L_t at given time-points $\{k\Delta t, k = 0, \ldots, n-1\}$.

Another useful property of the NIG process is the fact that if we model our underlying as

$$S_t = S_0 \exp(L_t)$$

with respect to the real-world measure $\mathbb{P}$, we can use the Esscher transform to construct an equivalent martingale measure $\mathbb{Q}$ (see [9] and [10]), for a first occurrence of this method, which preserves the NIG structure of the driving Lévy process.

Hence the probability change from $\mathbb{P}$ to $\mathbb{Q}$ can be completely performed by adjusting the parameters α, β, δ and μ of the NIG process, so that from a numerical point of view it makes no difference, if we are modeling under the real-world measure or the risk-neutral one.

5.3.1 Computation of optimal quantizers for the NIG distribution

Recall that we may rewrite the L^2-quantization problem of the NIG distribution as

$$D_N := \min_{x \in \mathbb{R}^N} \sum_{i=1}^{N} \int_{C_i(x)} (\xi - x_i)^2 f^{\mathrm{NIG}}(\xi) d\xi \tag{23}$$

where $\{C_i(x), i = 1, \ldots, N\}$ denotes a Voronoi partition induced by $x = (x_1, \ldots, x_N)$.

The fact, that in the one-dimensional setting the Voronoi cells $C_i(x)$ are just intervals in $\mathbb{R}$ and the uniqueness of the optimal quantizer due to the unimodality of f^{NIG}, makes the quantization problem in this case very straightforward to solve.

Now assume $x = (x_1, \ldots, x_N) \in \mathbb{R}^N$ to be ordered increasingly and denote by

$$x_{1/2} := -\infty, \quad x_{i \pm 1/2} := \frac{x_i + x_{i \pm 1}}{2} \quad \text{for } 2 \leq i \leq N-1, \quad x_{N+1/2} := +\infty$$

the midpoints between the quantizer elements respectively $\pm\infty$. A Voronoi partition of Γ is therefore given by

$$\begin{aligned} C_1(x) &= (-\infty, x_{1+1/2}], \\ C_i(x) &= (x_{i-1/2}, x_{i+1/2}], \quad 2 \leq i \leq N-1, \\ C_N(x) &= (x_{N-1/2}, +\infty). \end{aligned}$$

so that (23) now reads

$$\min_{x \in \mathbb{R}^N} \sum_{i=1}^{N} \int_{x_{i-1/2}}^{x_{i+1/2}} (\xi - x_i)^2 f^{\mathrm{NIG}}(\xi) d\xi. \tag{24}$$

This is an N-dimensional optimization problem, which can be easily solved by Newton's method as soon as we have access to the first and second order derivatives of D_N.

In fact, we can calculate the first order derivative of D_N (see e.g. [12], Lemma 4.10 or [19], Lemma C) as

$$\frac{\partial D_N}{\partial x_i}(x) = 2 \int_{x_{i-1/2}}^{x_{i+1/2}} (x_i - \xi)\ f^{\mathrm{NIG}}(\xi) d\xi, \quad 1 \leq i \leq N. \tag{25}$$

Moreover the Hessian matrix turns out to be a symmetric tridiagonal matrix with diagonal entries

$$\begin{aligned} \frac{\partial^2 D_N}{\partial x_1^2}(x) &= 2 \int_{x_{1/2}}^{x_{1+1/2}} f^{\mathrm{NIG}}(\xi) d\xi - \frac{x_2 - x_1}{2} f^{\mathrm{NIG}}(x_{1+1/2}) \\ \frac{\partial^2 D_N}{\partial x_i^2}(x) &= 2 \int_{x_{i-1/2}}^{x_{i+1/2}} f^{\mathrm{NIG}}(\xi) d\xi - \frac{x_i - x_{i-1}}{2} f^{\mathrm{NIG}}(x_{i-1/2}) - \frac{x_{i+1} - x_i}{2} f^{\mathrm{NIG}}(x_{i+1/2}), \quad 2 \leq i \leq N \\ \frac{\partial^2 D_N}{\partial x_N^2}(x) &= 2 \int_{x_{N-1/2}}^{x_{N+1/2}} f^{\mathrm{NIG}}(\xi) d\xi - \frac{x_N - x_{N-1}}{2} f^{\mathrm{NIG}}(x_{N-1/2}) \end{aligned} \tag{26}$$

and the super- respectively sub-diagonals

$$\frac{\partial^2 D_N}{\partial x_i \partial x_{i-1}}(x) = \frac{\partial^2 D_N}{\partial x_{i-1} \partial x_i}(x) = -\frac{x_i - x_{i-1}}{2} f^{\mathrm{NIG}}(x_{i-1/2}), \quad 2 \le i \le N.$$

As a consequence of (25), the remaining entries vanish

$$\frac{\partial^2 D_N}{\partial x_i \partial x_j}(x) = \frac{\partial^2 D_N}{\partial x_j \partial x_i}(x) = 0, \quad 1 \le i < j - 1 \le N - 1.$$

For the evaluation of the integrals occurring in the expressions (25) and (26) we employed high-order numerical integration methods which gave satisfying results.

But some attention should be paid to the initialization of the Newton's method, since it converges only locally. We achieved a fast convergence using an equidistant placed N-tuple in the interval $[\mathbb{E}L_{t_k} - 2\operatorname{Var} L_{t_k}, \mathbb{E}L_{t_k} + 2\operatorname{Var} L_{t_k}]$ as starting vector, i.e.

$$\left[t_k \left(\mu + \frac{\delta\beta}{\sqrt{\alpha^2 - \beta^2}} - 2\frac{\delta\alpha^2}{(\alpha^2 - \beta^2)^{3/2}} \right), \quad t_k \left(\mu + \frac{\delta\beta}{\sqrt{\alpha^2 - \beta^2}} + 2\frac{\delta\alpha^2}{(\alpha^2 - \beta^2)^{3/2}} \right) \right]$$

for the case of the NIG, so that we reached the stopping criterion of $\|\nabla D_N(x^*)\| \le 10^{-8}$ within $20 - 30$ iterations.

5.3.2 Simulation of ΔL_t

For the simulation of a $\mathrm{NIG}(\alpha, \beta, \delta, \mu)$ process it is useful to regard L_t as subordinated to a Brownian motion, i.e.

$$L_t = \beta\delta^2 I_t + \delta W_{I_t} + \mu t,$$

for an *Inverse Gaussian* process I_t, that is again a Lévy process with Inverse Gaussian distribution, $\mathrm{IG}(a, b)$, and parameters $a = t$ and $b = \sqrt{\alpha^2 - \beta^2}$.

A way to simulate the $\mathrm{IG}(a, b)$ distribution was proposed in [7], p.182.

6 Numerical results

In the above explained models we performed some tests on the pricing of swing options.

6.1 The pricing procedure

Given any initial global constraints $Q \in \mathbb{N}^2 \cap T^+(n)$, the quantized tree algorithm aims at backward solving the following program

$$\begin{aligned} \hat{P}_n^n &\equiv 0 \\ \hat{P}_k^n(Q^k) &= \max\left\{ x v_k(\hat{X}_k) + \mathbb{E}(\hat{P}_{k+1}^n(\chi^{n-k-1}(Q^k, x))|\hat{X}_k), x \in \{0,1\} \cap I_{Q^k}^{n-k-1} \right\}, \qquad k = 0, \dots, n-1, \end{aligned} \tag{27}$$

for all admissible Q^k from section 3.1. Here, v_k is a local payoff with strike K, which is defined by

$$v_k(x) = (F_{0,k\Delta t} \exp(\sigma x - \frac{1}{2}\Delta_{k\Delta t}^2) - K),$$

for the Gaussian 1-factor model,

$$v_k(x^1, x^2) = (F_{0,k\Delta t} \exp(\sigma_1 x^1 + \sigma_2 x^2 - \frac{1}{2}\Delta_{k\Delta t}^2) - K),$$

for the Gaussian 2-factor model and

$$v_k(x) = S_0 \exp(x) - K$$

for the NIG-model.

Note that the only admissible Q^0 is Q, and therefore $P_0^n(Q) := P_0^n(Q^0)$ gives then the premium of a swing option.

In all test cases, we set the local constraints to

$$q_{\min} = 0 \quad \text{and} \quad q_{\max} = 6$$

and give results for a number of exercise dates $n = 30$ resp. $n = 365$ and strikes

$$K = 5, 10, 15, 20.$$

As a benchmark served the call strip case

$$Q = (0,\, q_{\max} \cdot n),$$

which reduces in view of (9) to the normalized case $\tilde{Q} = (0, n)$ and therefore has a reference price given by a sum of plain vanilla calls (see section 3.3).

The tests were carried out in `JAVA` on a dual `Intel Xeon QuadCore` CPU@3GHz and the `CUDA`-results on a `NVIDIA GTX 280` GPU.

Moreover, the Monte-Carlo sample size were in all cases chosen as $M = 10^5$ per time layer t_k.

6.2 1-factor model

For the Gaussian 1-factor model we have chosen the parameters

$$\sigma = 0.7, \alpha = 4, F_{0,t} = 20$$

with respect to a 30-day period for the underlying dynamic. We give results for the diffusion method (diff), as representative for the MC-based approaches, and the deterministic spray method (dSpray), which uses a 63-point Gauss quadrature for the numerical integration.

In addition, we implemented the *trinomial tree* method from [14], which is based on a tree dynamics for a mean reverting process from [13]. In fact, this approach fits exactly into our pricing framework, where only the grid points and the 3-way transitions have to be chosen accordingly to [14], section 4.1.

Unfortunately, this design is only defined for a special number of grid points, which is 15 in our parameter setting. Hence, we could carry out comparisons to the trinomial tree method only for the case $N = 15$.

Concerning the accuracy of these methods, we start with the call strip-case in Table 1.

	$K = 10$								
N	diff	rel. Err.	sec.	dSpray	rel. Err.	sec.	trinom	rel. Err.	sec.
15	1799.02	0.073%	1.52	1797.54	0.155%	0.08	1808.06	0.430%	0.001
50	1800.37	0.003%	1.65	1799.97	0.020%	0.45			
100	1800.49	0.009%	1.74	1800.23	0.005%	1.65			
200	1800.51	0.010%	1.86	1800.30	0.001%	6.48			
	$K = 20$								
N	diff	rel. Err.	sec.	dSpray	rel. Err.	sec.	trinom	rel. Err.	sec.
15	319.58	0.210%	1.52	316.61	1.137%	0.08	327.36	2.218%	0.001
50	320.13	0.037%	1.65	319.57	0.211%	0.45			
100	320.38	0.041%	1.74	320.10	0.046%	1.65			
200	320.40	0.048%	1.86	320.21	0.013%	6.48			

Table 1: Gaussian 1-factor model for $n = 30$ and $Q = (0, 180)$ with computational times for the transition probabilities on a single CPU.

For both strikes $K = 10, 20$ the quantization methods induce already for $N = 15$ a much smaller approximation error than the trinomial tree method, since they are based on a discretization which is fitted in an optimal way to the underlying distribution, in contrast to the trinomial tree method which is based on an equidistant discretization.

Moreover, with a grid size of $N = 50$, both quantization methods provide relative errors in the region of a few ‰ or less, which is nearly the exact result.

Since the diffusion approach contains, compared to the spray method, one approximation less, it outperforms the deterministic spray method for smaller N as shown in the table.

In view of the computational time for the estimation of the transition probabilities, the trinomial tree approach, where each node can only take 3 different states, performs very fast.

Nevertheless, the deterministic spray method executes for $N = 15$ in less than 1/10 second and for $N = 50$ in less than 1/2 second, which is still instantaneous and provides moreover a much higher accuracy than the trinomial tree method.

Concerning MC-simulation based approaches, we here only give computation times for the diffusion approach, which is about $1\frac{1}{2}$ second and therefore cannot compete with the other approaches for small and moderate N.

Once the computation of the transition probabilities is done, the pricing method has to iteratively traverse the quantization tree for the solution of the stochastic control problem in (27), whose execution time are given in Table 2.

N	15	50	100	200
sec.	0.02	0.10	0.17	0.29

Table 2: Computational times on a single CPU for the traversal of the quantization tree in the 1-factor model.

These times, which are independent of the transition approach and which can be considered as the minor compute intensive problem, have to be added to those of Table 1 to arrive at the total time for the pricing of one contract.

To illustrate the performance of these methods in case of a non-trivial control problem, we chose global constraints

$$Q = (100, 150)$$

in Table 3.

	$K = 10$		
N	diff	dSpray	trinom
15	1589.97	1587.66	1602.48
50	1588.95	1588.41	
100	1588.89	1588.51	
200	1588.86	1588.54	
	$K = 20$		
N	diff	dSpray	trinom
15	226.83	223.76	241.29
50	225.28	224.75	
100	225.19	224.90	
200	225.15	224.94	

Table 3: Gaussian 1-factor model for $n = 30$ and $Q = (100, 150)$.

Here again, the quantization methods are very close to each other and suggest that the true price for $K = 20$ is around 225, which means that these methods again outperform the trinomial tree method by several orders of magnitude.

Moreover, the consistency with Table 1 justifies the use of the call strip case as a general benchmark for the performance.

In all these examples the dSpray-method with $N = 50$ performs the pricing of a 30-day contract in about 1/2 second and with an error of only a few ‰, which seems to be a good compromise between speed and accuracy.

6.3 2-factor model

For the numerical analysis in the Gaussian 2-factor model, we set parameters

$$\sigma_1 = 0.36, \alpha_1 = 0.21, \sigma_2 = 1.11, \alpha_2 = 5.4, \rho = -0.11, F_{0,t} = 20$$

with respect to a one year period.

Since the 2-factor model is based on bivariate normal distribution in each time layer, deterministic approaches are not competitive anymore, and we therefore focus only on the MC-based methods, i.e. the diffusion method (diff), the parallel Quantization Weight Estimation-method (pQWE) and the spray method for MC-simulation (MCSpray).

Starting with a 30-day contract in Table 4,

	$K = 10$								
N	diff	rel. Err.	sec.	pQWE	rel. Err.	sec.	MCSpray	rel. Err.	sec.
100	1796.57	0.202%	3.84	1798.26	0.108%	4.78	1793.11	0.394%	2.56
250	1797.58	0.146%	4.64	1799.91	0.016%	5.21	1794.72	0.305%	2.67
500	1797.94	0.126%	5.26	1800.94	0.041%	5.57	1795.98	0.235%	2.92
	$K = 20$								
N	diff	rel. Err.	sec.	pQWE	rel. Err.	sec.	MCSpray	rel. Err.	sec.
100	263.72	1.816%	3.84	264.38	1.568%	4.78	254.55	5.229%	2.56
250	265.50	1.153%	4.64	266.64	0.728%	5.21	259.64	3.334%	2.67
500	266.16	0.906%	5.26	267.70	0.334%	5.57	262.99	2.086%	2.92

Table 4: Gaussian 2-factor model for $n = 30$ and $Q = (0, 180)$ with computational times for the transition probabilities on a single CPU.

the diffusion and the pQWE-method perform similarly with respect to execution time and accuracy, whereas the spray method induces about double the error for half the execution time.

In this 2-factor setting we also present results for a 365-day contract in table 5.

	$K = 10$								
N	diff	rel. Err.	sec.	pQWE	rel. Err.	sec.	MCSpray	rel. Err.	sec.
100	21943.77	0.662%	88	21960.16	0.588%	58	21758.67	1.500%	22
250	22006.16	0.379%	118	22001.90	0.399%	62	21580.61	2.306%	24
500	22042.31	0.216%	145	22052.94	0.168%	67	21756.86	1.508%	27
	$K = 20$								
N	diff	rel. Err.	sec.	pQWE	rel. Err.	sec.	MCSpray	rel. Err.	sec.
100	6398.35	2.084%	88	6380.18	2.363%	58	6090.92	6.789%	22
250	6470.05	0.987%	118	6437.34	1.488%	62	5926.04	9.312%	24
500	6508.88	0.393%	145	6483.65	0.779%	67	6100.28	6.646%	27

Table 5: Gaussian 2-factor model for $n = 365$ and $Q = (0, 2190)$ with computational times for the transition probabilities on a single CPU.

Here we observe execution time in the range of about 1/2 minute up to 2 minutes for the single core implementation, which becomes quite critical for time sensitive applications.

6.3.1 Parallel implementation

To demonstrate the parallel performance of the transition estimation methods, we first implemented the pQWE-methods using 365 threads (for each time layer one) on 8 Xeon cores. This approach reduces the execution to about 7 seconds (cf. Table 6), which means that it scales linearly with respect to number of available processors (and even a bit better, since these 365 threads exploit the computing power of a single core more efficiently than a single thread).

Moreover, we developed a `CUDA`-implementation of the pQWE-method, which contains beneath the layer-wise parallelization also a MC-wise one, i.e. 365 blocks with each containing 256 threads for the MC-simulation per layer.

This finally enabled us to price the 365-day contract at $N = 100$ in less than 1 second with about 2% accuracy using the 240 ScalarProcessors of an `NVIDIA GTX 280` GPU.

N	100	250	500
$8 \times$ `Xeon`	7.12	7.33	7.79
`GTX 280`	0.75	1.26	2.11

Table 6: Computational time in seconds for pQWE-method on the 2-factor model with $n = 365$.

Note, that the dual `Xeon` has an overall computation power of about 192 GFLOPS, whereas the `GTX 280` can perform up to 933 GFLOPS.

6.3.2 Romberg extrapolation

A further way to improve the accuracy of the quantization method is the extrapolation method from section 4.3 here applied to the 30-day contract from Table 4.

	$K = 5$					
N	diff	rel. Err.	pQWE	rel. Err.	MCSpray	rel. Err.
$100 - 250$	2698.04	0.073%	2700.79	0.029%	2695.64	0.162%
$250 - 500$	2698.09	0.071%	2701.77	0.065%	2697.09	0.108%
	$K = 10$					
N	diff	rel. Err.	pQWE	rel. Err.	MCSpray	rel. Err.
$100 - 250$	1798.25	0.109%	1801.01	0.045%	1795.79	0.246%
$250 - 500$	1798.30	0.106%	1801.97	0.098%	1797.24	0.165%
	$K = 15$					
N	diff	rel. Err.	pQWE	rel. Err.	MCSpray	rel. Err.
$100 - 250$	922.31	0.234%	924.79	0.035%	918.73	0.624%
$250 - 500$	922.43	0.220%	925.92	0.157%	921.23	0.351%
	$K = 20$					
N	diff	rel. Err.	pQWE	rel. Err.	MCSpray	rel. Err.
$100 - 250$	266.68	0.716%	268.14	0.168%	263.03	2.114%
$250 - 500$	266.82	0.663%	268.75	0.060%	266.34	0.844%

Table 7: Extrapolation for Gaussian 2-factor model with $n = 30$ and $Q = (0, 180)$

Here, we observe for all the methods a boost up of the accuracy, which in particular holds for the pQWE-approach.

6.4 NIG-model

To finally apply the method in a non-Gaussian setting, we tested the exponential Lévy-model from section 5.3 for daily parameters

$$\alpha = 50, \quad \beta = -2.0, \quad \delta = 0.02, \quad \mu = 0.001, \quad s_0 = 20$$

and a 30 days contract.

Here again as in the 1-factor model (cf. Table 1), the dSpray-method performs very well and even outperforms in this case the diffusion method. Moreover it reveals a very stable extrapolation behavior, which can be seen in table 9.

Since the computational times agreed with those from the 1-factor setting, we did not reproduce them again.

	$K = 10$			
N	diff	rel. Err.	dSpray	rel. Err.
50	1820.22	0.040%	1820.59	0.020%
100	1820.23	0.039%	1820.90	0.003%
200	1820.24	0.039%	1820.94	$< 0.001\%$
	$K = 20$			
N	diff	rel. Err.	dSpray	rel. Err.
50	111.92	0.356%	111.93	0.347%
100	111.96	0.321%	112.22	0.088%
200	111.97	0.313%	112.30	0.022%

Table 8: NIG-model with $n = 30$ and $Q = (0, 180)$

	$K = 10$			
N	diff	rel. Err.	dSpray	rel. Err.
$50 - 100$	1820.23	0.039%	1821.00	0.003%
$100 - 200$	1820.24	0.039%	1820.95	$< 0.001\%$
	$K = 20$			
N	diff	rel. Err.	dSpray	rel. Err.
$50 - 100$	111.97	0.311%	112.32	0.002%
$100 - 200$	111.97	0.312%	112.32	$< 0.001\%$

Table 9: Extrapolation for NIG-model with $n = 30$ and $Q = (0, 180)$

7 Conclusions

We have systematically carried out numerical tests on the pricing of swing options in the quantization tree framework using different methods for the transition probability estimation.

The results are very promising, since we could show that

- already in the 1-factor model the quantization approach outperforms the trinomial tree methods due to its better fit to the underlying distribution,
- the computational time for the transition probability estimation in the 2-factor model with 365 exercise dates can be reduced to less than 1 second using our parallel approaches on a nowadays GPU device,
- the quantization framework also works very well for non-Gaussian, in our case a NIG Lévy process, Markov Feller underlyings.

Moreover, we could speed up the convergence of all these approaches by means of a Romberg extrapolation.

References

[1] V. Bally, G. Pagès, and J. Printems. A quantization tree method for pricing and hedging multidimensional American options. *Math. Finance*, 15(1):119–168, 2005.

[2] O. Bardou, S. Bouthemy, and G Pagès. When are Swing options bang-bang and how to use it. LPMA preprint, 2007.

[3] O. Bardou, S. Bouthemy, and G Pagès. Optimal Quantization for the Pricing of Swing Options. *Applied Mathematical Finance*, 16(2):183–217, 2009.

[4] O. E. Barndorff-Nielsen. Processes of normal inverse Gaussian type. *Finance Stoch.*, 2(1):41–68, 1998.

[5] F.E. Benth, D. Frestad, and S. Koekebakker. Modeling the term structure dynamics in the Nordic Electricity swap market. Preprint, 2007.

[6] F.E. Benth and J. Saltyte-Benth. The normal inverse Gaussian distribution and spot price modeling in energy markets. *Intern. J. Theor. Appl. FinanceJournal*, 7(2):177 – 192, 2004.

[7] R. Cont and P. Tankov. *Financial modelling with jump processes.* Financial Mathematics Series. Chapman & Hall, 2004.

[8] J. H. Freidman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.*, 3(3):209–226, 1977.

[9] H.U. Gerber and E.S.W. Shiu. Option pricing by Esscher transforms. *Trans. Soc. Actuaries*, 46:99–191, 1994.

[10] H.U. Gerber and E.S.W. Shiu. Actuarial Approach to Option Pricing. In *Proceedings of the 5th AFIR International Colloquium*, volume 1, pages 43–96, 1995.

[11] A. Gersho and R.M. Gray. *Vector Quantization and Signal Compression.* Kluwer, Boston, 1992.

[12] S. Graf and H. Luschgy. *Foundations of Quantization for Probability Distributions.* Lecture Notes in Mathematics $n^0$1730. Springer, Berlin, 2000.

[13] A. Hull, J. White. Numerical procedures for implementing term structure models I: Single-Factor Models. *Journal of Derivatives*, 2(1):7–16, 1994.

[14] P. Jaillet, E. I. Ronn, and S. Tompaidis. Valuation of Commodity-Based Swing Options. *Managment Science*, 50(7):909–921, 2004.

[15] J. C. Kieffer. Uniqueness of locally optimal quantizer for log-concave density and convex error weighting function. *IEEE Trans. Inform. Theory*, 29(1):42–47, 1983.

[16] G. Pagès. A space vector quantization method for numerical integration. *Journal of Applied and Computational Mathematics*, 89:1–38, 1997.

[17] G. Pagès and J. Printems. Optimal quadratic quantization for numerics: the gaussian case. *Monte Carlo Methods and Applications*, 9(2):135–166, 2003.

[18] G. Pagès and J. Printems. Optimal Quantization for Finance: From Random Vectors to Stochastic Processes. In P.G. Ciarlet, editor, *Handbook of Numerical Analysis*, volume XV of *Mathematical Modeling and Numerical Methods in Finance*, pages 595–648. North-Holland, 2009.

[19] D. Pollard. A central limit theorem for k-means clustering. *Ann. Probab.*, 10(4):919–926, 1982.